User Guide:
Client Feed API

Document Creation Date:	15 Feb 2012
Document Version:	2.4
API version:	1
Author:	Rob Steadman
Contributors:	Simon George
	Cathy Pollitt

Copyright

All contents © copyright GMG Property Services 2010-2012.

All rights reserved. No parts of this document or the related files may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without the prior written permission of the publisher.

Document History

Date *(dd mmm yyyy)*	***Version*** *(n.nn)*	***Author***	***Reason for Change*** *(including paragraph references)*
23 Sep 2010	1.0	Rob Steadman	First Issue
08 Nov 2010	1.1	Cathy Pollitt	Revisions following review ready for Beta release: • Changed format for display of HTTP Statuses for readability • Glossary of Terms updated • Added copyright information • Expanded Schema section
15 Nov 2010	1.2	Simon George	Re-written and re-ordered.
04 Mar 2011	1.3	Rob Steadman	Revisions to Authentication
7 Mar 2011	2.0	Simon George	API Versioning added
29 June 2011	2.1	Cathy Pollitt	Minor revisions following review
01 July 2011	2.2	Simon George	Minor revisions following review
15 Aug 2011	2.3	Rob Steadman	Revision to Authentication Section
15 Feb 2012	2.4	Cathy Pollitt	1. Added API Version to Title page 2. Changed section title from “Authentication” to “Authentication and Use of Tokens” 3. Expanded Purpose of Document section 4. Updated API Calls section relating to the Get Changed Files call 5. Updated API Call List / Get Changed Properties section to clarify the updates returned 6. Updated API Call List / Get Changed Files section to clarify the updates returned and limitations of the call in the current version 7. Expanded HTTP Status Codes section 8. Expanded Glossary of Terms section 9. Updated copyright

Table of Contents

COPYRIGHT **1**
DOCUMENT HISTORY **2**
TABLE OF CONTENTS **3**
PURPOSE OF DOCUMENT **4**
SUMMARY OF API FUNCTIONALITY **4**
YOUR DATA FEED IDENTIFIER **4**
AUTHENTICATION AND USE OF TOKENS **4**
API CALLS **6**
API VERSIONS **6**
CACHING **6**
SCHEMA **7**
HOW TO USE THE API **8**
Initial Population of your database 8
On-going Real-time updates 8
API CALL LIST **9**
Get Branches 9
Get Branch 9
Get Property List 10
Get Property 10
Get Changed Properties 11
Get Changed Files 12
HTTP STATUS CODES **13**
GLOSSARY OF TERMS **14**

Purpose of Document

1. The purpose of this document is to provide a technical user guide to the GMGPSG Client Feed API.
2. Although this document contains technical information, all attempts have been made to use plain English throughout.
3. The intended audience of this document are clients of GMG Property Services Group (GMGPSG) and their web developers.
4. The accompanying PHP script is provided as an example of consuming the API in PHP. We do not and cannot provide technical support on implementing a PHP solution to developers.

Summary of API Functionality

The Client Feed API provides a method by which clients of GMG Property Services Group are able to retrieve their data in real time in a standardized format so it can be stored in your database to power your website. This is delivered via a web XML API.

We will support multiple versions of the API. We will notify you when new versions of the API become available.

Your data feed identifier

You have been given a data feed identifier. This identifier is associated with the branch or branches for which you will be retrieving data. You will use this identifier in all API calls to the service where you see {datafeedid} in the URL. Use your identifier in place of {datafeedid} with no whitespace before or after it. The data feed identifier is not case sensitive.

Authentication and Use of Tokens

API calls must be authenticated using HTTP Basic Authentication.

You have been given a username and password for your data feed - these are case sensitive. You will need to add an 'Authorization' header to the request, containing the string "username:password" (encoded to base 64), with the word "Basic " (plus space) in front.

The response to your initial request will contain a security token (accessed by the name 'Token' in the header) which must replace the 'username:password' string in subsequent requests. The token is valid for one hour and **must** be used until it expires – when it does, a 401 (unauthorized) will be returned and another token must be requested, using the username & password again.

C# Asp.net Example 1:

```
HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);

string username = "user";
string password = "password";

string usernamePassword = username + ":" + password;
```

```
CredentialCache cache = new CredentialCache();

cache.Add(new Uri(url), "Basic", new NetworkCredential(username,
 password));
request.Credentials = cache;
request.Headers.Add("Authorization", "Basic " // <- space here.
 + Convert.ToBase64String(new ASCIIEncoding().GetBytes(usernamePassword)));

// Get the token from the response:
string token = response.GetResponseHeader("Token");
```

C# Asp.net Example 2 - using the token in subsequent requests:

```
HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);

string token = "token1";

CredentialCache cache = new CredentialCache();

cache.Add(new Uri(url), "Basic", new NetworkCredential(username,
 password));
request.Credentials = cache;
// Use token here instead of username:password string.
request.Headers.Add("Authorization", "Basic " // <- space here.
 + Convert.ToBase64String(new ASCIIEncoding().GetBytes(token)));
```

API Calls

There are six API calls.

- Get Branches – returns a list of all branches
- Get Branch – returns more details for a branch
- Get Property List – returns list of properties for a branch
- Get Property – returns the property details
- Get Changed Properties – returns a list of properties that have changed (updated, added or deleted) since a certain date and time.
- Get Changed Files – returns a list of files that have changed (updated or added) since a certain date and time.

These API calls must all be authenticated.

All the API calls return XML. This XML is defined by a schema. This schema documents the XML returned.

API Versions

The API supports multiple versions of the xml and appropriate schema, allowing GMGPSG to add data and functionality to the API without affecting existing users.

We will notify you when new versions of the Client Feed API are available.

In line with Google, Twitter and Facebook, older versions of the API will be deprecated in time and sufficient notice will be given to migrate to the latest version.

The API version is part of the URL on all API calls. All API URLs start with

http://webservices.vebra.com/export/{datafeedid}/v{version}

{version} is the version required. The current version is 1.

Caching

The Branch, Branches, Property and Property List URLs implement the Last-Modified header to allow caching by the client. If the client sets the If-Modified-Since header in the request, only records modified since that date and time will be returned.

Schema

Each version of the API will have a corresponding schema and the XML shall reflect this.

For example;

API version 1 : *http://webservices.vebra.com/export/xsd/v1/exportapi.xsd*

The XML returned will include the namespace in the root node, e.g.

<branch xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xsi:noNamespaceSchemaLocation=”http://webservices.vebra.com/export/xsd/v1/ exportapi.xsd”>

The following schema files are also referenced:

Property Export: *http://webservices.vebra.com/export/xsd/v1/propertyexport.xsd*

Currency: *http://webservices.vebra.com/export/xsd/v1/currency.xsd*

Country, Market and Category: *http://webservices.vebra.com/export/xsd/ v1/dbids.xsd*

Property Related Types: *http://webservices.vebra.com/export/xsd/v1/propertyrelatedtypes.xsd*

Rightmove Types: *http://webservices.vebra.com/export/xsd/v1/rightmovetypes.xsd*

Simple Types: *http://webservices.vebra.com/export/xsd/v1/simpletypes.xsd*

How to use the API

This section provides guidance on how you would use the API to initially populate your database and keep your data in sync, in near to real-time.

Initial Population of your database

1. Call 'Get Branches' – this returns XML with a list of branches and a URL to the branch details for each branch.

2. For each branch in that list, call 'Get Branch' and store the branch details in your database. (It would be useful to store the URL and date and time you queried this branch. Then you can periodically check this and use the date and time in the 'If-Modified-Since' header. If the data has changed you will get the changed data, otherwise the API will return 304 Not-modified. This can help you reduce the volume of database updates.)

3. For each branch in that list, call 'Get Property List'

4. For each property that 'Get Property List' returns, call 'Get Property' (the URL returned by 'Get Property List') and store the properties locally.

5. For each property you should download the image(s) and referenced file(s) and store them.

6. It would be sensible to allow your system to be able repeat the 'initial population of your database' if you lose sync.

On-going Real-time updates

Periodically you should use the API Calls detailed below to get changed properties and files. To do this:-

1. Call 'Get Changed Properties' using the date and time you last checked for updates / initially populated your database.

2. For each property returned, perform the appropriate operation (get property then update, or insert or remove the property) to your database.

3. Store the date and time you checked for updates for the next time you call the API.

4. Repeat the same process for files using 'Get Changed Files'

API Call List

This section details the list of calls available.

Get Branches

http://webservices.vebra.com/export/{datafeedid}/v{version}/branch

{datafeedid} is the identifier for this data feed.
{version} the current version is 1.

Returns XML with list of branches for the data feed containing

- branch name
- firmid
- branchid
- URL to full branch XML ('Get Branch' call)

Please see the schema for full details.

Example URL:

http://webservices.vebra.com/export/homeestates/v1/branch

(This URL is not active)

Get Branch

http://webservices.vebra.com/export/{datafeedid}/ v{version}/branch/{clientid}

{datafeedid} is the identifier for this data feed.
{version} the current version is 1.
{clientid} is the branch identifier.

This URL is present in the 'Get Branches' XML for each branch.

Please see the schema for full details of the XML returned.

Example URL:

http://webservices.vebra.com/export/ homeestates /v1/branch/1234

(This URL is not active)

Get Property List

http://webservices.vebra.com/export/{datafeedid}/ v{version}/branch/{clientid}/property

{datafeedid} is the identifier for this data feed.
{version} the current version is 1.
{clientid} is the branch identifier.

BETA: use the 'Get Branch' URL and append '/property' to make this API call.

Returns XML with the list of properties for the branch specified in the {clientid} containing (for each property)

- prop_id (the property identifier)
- last changed date and
- URL to full property details XML ('Get Property' call).

Please see the schema for full details of the XML returned.

Example URL:

http://webservices.vebra.com/export/ homeestates /v1/branch/1234/property

(This URL is not active)

Get Property

http://webservices.vebra.com/export/{datafeedid}/ v{version}/branch/{clientid}/property/{prop_id}

{datafeedid} is the identifier for this data feed.
{version} the current version is 1.
{clientid} is the branch identifier.
{prop_id} is the unique property identifier.

Returns XML with the full property details for the property specified in {prop_id}

Please see the schema for full details of the XML returned.

Example URL:

http://webservices.vebra.com/export/ homeestates / v1/branch/ 1234/property/12345678

(This URL is not active)

Get Changed Properties

http://webservices.vebra.com/export/{datafeedid}/v{version}/property/{yyyy}/{MM}/{dd}/{HH}/{mm}/{ss}

{datafeedid} is the identifier for this data feed.
{version} the current version is 1.

This call expects a date and time after which property changes are returned.

- {yyyy} – year
- {MM} – month with leading zeros
- {dd} – day of the month with leading zeros
- {HH} – the hour of the day (0 – 23) with leading zeros
- {mm} – minutes with leading zeros
- {ss} – seconds with leading zeros

Returns XML with a list of properties for the data feed that have been updated or deleted since the date and time specified, containing (for each property)

- prop_id
- last changed date
- last action on property, and
- URL to full property XML

Valid Values for Last Action on property are:

- updated (this refers to a new property record or an update to an existing property record)
- deleted

Example URL:

http://webservices.vebra.com/export/ homeestates /v1/property/2010/08/31/01/01/01

(This URL is not active)

Note 1: The Get Changed Properties call returns results where the property record or its associated files or rooms have changed with a date/time equal to or greater than the date / time in the call.

Note 2: The ***lastchanged*** value in the returned results may sometimes have an earlier date/time than the call was made for. This will occur if the change related to the Rooms or Files associated with the property record that were equal to or greater than the date/time in the call but the property record itself has not changed.

Get Changed Files

http://webservices.vebra.com/export/{datafeedid}/v{version}/
files/{yyyy}/{MM}/{dd}/{HH}/{mm}/{ss}

{datafeedid} is the identifier for this data feed.
{version} the current version is 1.

This call expects a date and time after which property changes are returned.

- {yyyy} – year
- {MM} – month with leading zeros
- {dd} – day of the month with leading zeros
- {HH} – the hour of the day (0 – 23) with leading zeros
- {mm} – minutes with leading zeros
- {ss} – seconds with leading zeros

Returns XML with list of changed files for the feed containing

- file element with file_propid,
- last changed date
- URL to full property XML ('Get Property' call)
- URL of the file.

Example URL:

http://webservices.vebra.com/export/ homeestates /v1/files/2010/08/31/14/23/01

(This URL is not active)

Note 1: Deleted files are not supported in version 1 of the API and the Get Changed Files call will not return deleted files.

Note 2: Some file types are not currently supported in version 1 of the API for the Get Changed Files call (e.g. vebra360 and EPC) but they are returned in the Get Property call. This will be addressed in a future release.

HTTP Status codes

The following status codes will be returned as appropriate:

Status	***Description***
200 – OK	Returning the XML document
204 – No Content	Returned if the branch for which the Get Property List request was made has no properties
304 – Not Modified	The 'If-modified-since' header was supplied and content has not been changed since
401 – Unauthorized	Returned if the authentication fails
403 – Forbidden	Returned if the client is authenticated but tries to access a URL that is not available to the client (i.e. another data feed, or property that is not in the your data feed)
404 – Not Found	Returned for an incorrectly formed URL
410 – Gone	The property/branch has been deleted OR is not currently available.
500 – Internal Server Error	Something bad happened to the API.
501 – Not Implemented	Returned if the API version in the request is not supported (deprecated or unreleased versions)

Glossary of Terms

Term or Abbreviation	*Meaning*
API	**Application Programming Interface** A description of the way one piece of software asks another program to perform a service.
HTTP	**Hypertext Transfer Protocol** A networking protocol for distributed, collaborative, hypermedia information systems.
PHP	**Hypertext Preprocessor** Server-side HTML embedded scripting language.
URL	**Uniform Resource Locator** A pointer to a "resource" on the World Wide Web. A resource can be something as simple as a file or a directory, or it can be a reference to a more complicated object, such as a query to a database or to a search engine.
XML	**Extensible Markup Language** A set of rules for encoding documents in machine-readable form